Introdução a Lógica de Programação

APRESENTAÇÃO

A base fundamental de um programa de computador são os algoritmos e a lógica de programação. Esses conceitos, além de serem bases da programação de computadores, desenvolvem habilidades, como raciocínio lógico, que podem ser utilizadas na resolução dos mais diversos problemas das mais diversas áreas.

Nesta Unidade de Aprendizagem, você estudará a construção de um algoritmo, os conceitos básicos envolvidos e como funciona a execução de um programa de computador.

Bons estudos.

Ao final desta Unidade de Aprendizagem, você deve apresentar os seguintes aprendizados:

- Conhecer o conceito de algoritmo: apresentação, exemplos e definição.
- Entender os conceitos de comandos, variáveis, expressões e instruções.
- Compreender as formas de execução de um programa: compilação e interpretação.

INFOGRÁFICO

Sabendo que um algoritmo nada mais é do que um conjunto de passos para se realizar uma tarefa, é importante compreender que não existe uma forma única de algoritmo, pois há várias formas de se cumprir um mesmo objetivo.

Acompanhe no infográfico como desenvolver um algoritmo.

Como desenvolver um algoritmo?

ENTENDER O PROBLEMA

Para desenvolver um algoritmo eficiente, é preciso entender muito bem o problema:

Quais informações já se tem?

O que se espera obter com esse algoritmo?

O que é possível fazer com as informações que se tem para conseguir chegar no objetivo?

PROJETAR

Nesta etapa, é preciso usar os conceitos básicos e padronizados para representar o algoritmo. É nela que se usam:

Variáveis

↓

Comandos

↓

Operações

↓

Estruturas de controle e/ou repetição

A representação pode ser textual ou por diagrama.

TESTAR

Esta é a etapa de teste do que foi projetado, ou seja, testa-se o algoritmo.

Ao final do teste, o valor esperado e o valor obtido devem ser iguais.

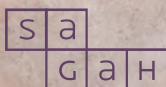
Caso esses valores sejam diferentes, volta-se para a etapa Analisar e/ou Projetar.

CONTEÚDO DO LIVRO

Aprender a programar é uma das habilidades mais interessantes da computação. Para que esse aprendizado seja consistente, é importante compreender toda a lógica que envolve o desenvolvimento de um programa: lógica de programação, conceitos básicos e execução de um programa são tópicos que introduzirão você ao universo da programação.

Leia o capítulo Algoritmos e lógica de programação, da obra *Algoritmos de programação*, base teórica para esta Unidade de Aprendizagem.

Boa leitura.

ALGORITMOS DE PROGRAMAÇÃO

Marcela Santos

Introdução à lógica de programação

Objetivos de aprendizagem

Ao final deste texto, você deve apresentar os seguintes aprendizados:

- Conhecer o conceito de algoritmo: apresentação, exemplos e definição.
- Entender os conceitos de comandos, variáveis, expressões e instruções.
- Compreender as formas de execução de um programa: compilação e interpretação.

Introdução

Já parou para pensar como é feito um jogo, um *site* da Internet ou um aplicativo do seu dispositivo móvel? Cada vez mais, usamos tecnologia desde o nosso trabalho até a nossa diversão. E que tal você aprender como os programas são desenvolvidos para poder criar seus próprios jogos, *sites*, etc.? A base fundamental de um programa de computador são os algoritmos e a lógica de programação. Esses conceitos, além de serem bases da programação de computadores, desenvolvem habilidades, como raciocínio lógico, que podem ser utilizadas na resolução dos mais diversos problemas nas mais diversas áreas.

Neste capítulo, você vai acompanhar a construção de um algoritmo, os conceitos básicos envolvidos e como funciona a execução de um programa de computador.

Algoritmo: apresentação, exemplos e definição

Um algoritmo nada mais é do que um conjunto de passos para a realização de uma determinada tarefa. O que é importante entendermos é que um algoritmo não precisa ser necessariamente algo ligado à tecnologia. Na realidade, o tempo todo estamos criando algoritmos para realizarem diversas tarefas. Quer ver um exemplo?

Quando você vai sair de casa a caminho da faculdade, existe uma sequência de passos que você realiza, que tem uma ordem. As ordens das coisas que você precisa fazer para chegar à faculdade são escolhidas para te ajudar a realizar a tarefa: “ir para a faculdade”.

Outros exemplos:

- Preparar um hambúrguer.
- Fazer um avião de papel.
- Trocar o pneu de um carro.

Exemplo

Exemplo — Preparar um hambúrguer

1. Cortar o pão e deixá-lo aberto com as partes de dentro voltadas para cima;
2. lavar a alface e o tomate;
3. fritar o hambúrguer;
4. colocar o hambúrguer sobre uma das partes do pão que está aberta;
5. colocar uma fatia de queijo, uma rodela de tomate e uma folha de alface sobre o hambúrguer;
6. fechar o pão com a outra parte que ficou sem nada sobre ela.

Pronto — sanduíche prontinho! Mas podemos fazer algumas perguntas sobre essa nossa aventura na cozinha:

- Existe somente um jeito de fazer esse hambúrguer?
- A ordem das atividades é importante?

Não existe uma única forma; o importante é que essas atividades sejam organizadas de tal forma que a tarefa “preparar um hambúrguer” seja realizada. Nesse momento, o importante é fazer, não importando se mais rápido, mais gostoso ou com mais recheio. Assim, a ordem é de extrema importância para que consigamos chegar ao objetivo final.

Afinal, o que isso tem a ver com computador? Bom, primeiro é preciso entender por que o uso de computador para resolução dos mais diversos problemas e das mais diversas tarefas tornou-se tão imprescindível. Para isso, vamos comparar o ser humano a um computador.

O humano é um ser inteligente, capaz de realizar diversas tarefas, como, por exemplo, calcular o valor do desconto de um jogo que está prestes a comprar. Já um computador é uma máquina "burrinha", que só executa o que mandamos. Em contrapartida, ele não se cansa. Imagina se você tivesse que calcular, durante o dia inteiro, o desconto da venda de uma loja? Poderia cansar-se e até cometer um erro, não é mesmo?

Então, é por isso que, desde quando foi criado até hoje, o papel fundamental do computador é ajudar o ser humano nas suas tarefas. Mas quem ensina o computador o que ele tem que fazer? Esse papel é do ser humano, que o faz por meio dos programas.

Os programas de computador são conjuntos de instruções que o computador precisa realizar para poder concluir uma determinada tarefa. Então, um algoritmo, em computação, também é um conjunto de passos, só que o foco é desenvolver um programa de computador.

Um algoritmo bastante famoso é o utilizado pelo Google, para realizar suas buscas. Ele foi desenvolvido a partir de 1995, pelos então estudantes Larry Page e Sergey Brin. Imagine que você tenha que realizar uma busca por nome em uma agenda telefônica — fica mais fácil se a agenda estiver organizada por nome, concorda? Então, os primeiros sistemas de busca ordenavam os *sites* dessa forma, sendo que o grande diferencial do Google foi utilizar um algoritmo que ordenasse os *sites* do mais acessado para os menos acessados. Dessa forma, a probabilidade do que o usuário estava procurando estar entre os primeiros é maior.

Os algoritmos estão em várias áreas — medicina, bolsa de valores, mobilidade urbana. Seu uso está cada vez mais difundido, e aprender como escrever um algoritmo bem como torná-lo mais eficiente é uma habilidade muito importante para os profissionais das mais diversas áreas.

Fique atento

Algoritmo pode ser definido como uma sequência de passos que visam a atingir um objetivo definido (FORBELLONE; EBERSPÄCHER, 2005).

Comandos, variáveis, expressões e instruções

No nosso exemplo do hambúrguer, falta pensarmos em uma coisa: saber como escrever e definir quais são os passos necessários. Bom, cada pessoa pode definir de um jeito, e isso poderia acontecer também ao escrevermos um algoritmo para um programa de computador, o que tornaria o trabalho dos computadores muito mais complicado, pois cada pessoa poderia definir esses passos de uma forma. Em computação, costuma-se padronizar grande parte dos conceitos envolvidos, e a definição dos passos de um algoritmo também foi padronizada.

Por meio de comandos, podemos formar as instruções e as expressões. Assim, os comandos são os tijolos da nossa construção; com eles, o computador entende o que deve fazer. Escrever um texto na tela, por exemplo, pode ser a ação que será executada pelo comando "escrever <texto" que será escrito na tela.

Existem vários tipos de comandos: comandos para atribuição de valor, comandos para entrada e saída de dados, comandos para estruturas de seleção ou de repetição — vamos conhecer todos eles. Agora, você precisa saber que comando denota uma ação (ou um conjunto delas) que o computador irá executar, tem forma e segue algumas regrinhas.

Outra definição que usaremos ao longo de todo o nosso curso é a de variáveis. Durante a execução de um programa, dados são entregues ao computador, que, executando esse programa, modifica os dados e "responde" conforme o esperado.

Exemplo

Exemplo — Operação de saque em um autoatendimento

1. Entrar com os dados de conta e senha;
2. solicitar qual opção o cliente deseja realizar;
3. se for saque, solicitar o valor a ser sacado;
4. se existir saldo, entregar o dinheiro;
5. se não, mostrar a mensagem de "saldo indisponível" na tela;
6. encerrar a operação.

Esse valor que você solicitou é um dado que precisa ser armazenado na memória do computador. Essa região da memória do computador, onde guardamos, mesmo que temporariamente, o dado, chamamos de "variável", e, como o próprio nome diz, pode sofrer mudança (variação) de valor.

Saiba mais

Memória de um computador é o meio físico para armazenar dados temporariamente ou permanentemente. Para saber mais, consulte TANENBAUM, 2000.

Para podermos usar as variáveis, precisamos inicialmente criá-las. Em programação, a criação de uma variável é chamada de "declaração de uma variável", que pode ser comparada ao ato de etiquetar pequenas gavetas para organizar diversos itens. A grande ideia por detrás da variável é que cada uma delas tem um nome e um tipo. Assim, somente os dados daquele tipo podem ser colocados na sua respectiva variável. Não daria para colocar um grampeador em uma gaveta etiquetada com canetas. Veremos mais detalhes sobre variável adiante.

Com as variáveis e os comandos, podemos formar as instruções e/ou expressões. Uma expressão é uma combinação de variáveis, comandos e operadores. Já uma instrução é uma operação única, que é executada pelo processador do computador. Assim, o programa de computador é formado por várias instruções. Em resumo, as instruções são formadas por comandos e/ou variáveis. Todas essas ferramentas são usadas para escrever um programa, utilizando o conceito de algoritmo.

Representação de um algoritmo

Como foi visto, um algoritmo é um conjunto de passos para que uma tarefa seja realizada. Esses passos são organizados de forma lógica, a fim de se chegar ao objetivo final. Existem várias formas de se representar um algoritmo — até agora usamos o português para isso, mas existem representações gráficas: fluxograma e diagrama de Chapin. Vamos representar o algoritmo do saque, usando o fluxograma da Figura 1:

INÍCIO

Solicitar dados do usuário

Solicitar a opção de operação

Se for saque

V

Solicitar valor a ser sacado

F

Saldo suficiente?

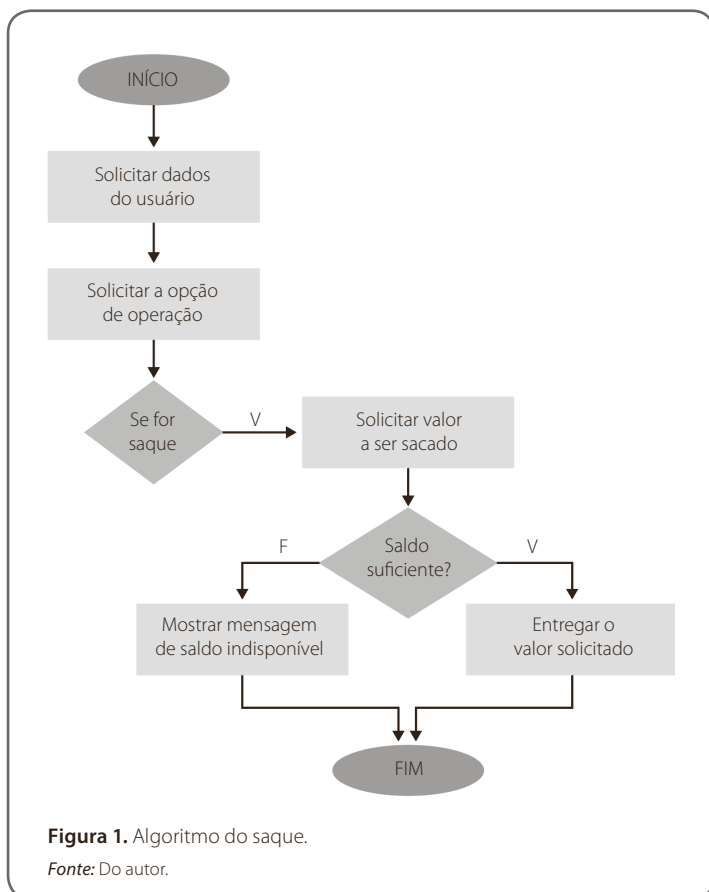
V

Mostrar mensagem de saldo indisponível

Entregar o valor solicitado

FIM

Figura 1. Algoritmo do saque.

Fonte: Do autor.

As representações gráficas são mais fáceis de entender devido à sua própria natureza. A lógica envolvida fica mais clara, porém é preciso que se aprenda a padronização dos itens que podem ser usados. Por esse motivo, ao logo deste livro, usaremos o português para a representação dos algoritmos.

Saiba mais

Para saber mais sobre a representação de algoritmos, consulte FORBELLONE; EBERSPÄCHER, 2005.

Execução de um programa: compilação e interpretação

Bom, já sabemos que um programa é um conjunto de instruções que são entregues ao computador para que ele realize determinada tarefa. Essa ação de realizar uma tarefa em computação chamamos de “execução de um programa”, e existem duas maneiras de um programa ser executado. Antes de passarmos para essa etapa, é importante dizer que os algoritmos, mesmo que padronizados, ainda não estão escritos em uma linguagem que o computador entenda. Existem várias linguagens que o computador entende: C, Java, Python são exemplos de linguagens de programação. Então, o processo de desenvolver um programa passa por estas etapas:

1. Entender o problema ou a tarefa que queremos que o programa execute.
2. Desenvolver um algoritmo.
3. Escrever o algoritmo em uma linguagem de programação — esse arquivo com o algoritmo escrito em uma linguagem de programação é chamado de “código-fonte”.
4. Enviar o programa para que o processador o execute, compilando ou interpretando o código-fonte.
   - Compilar um arquivo é traduzir de uma linguagem para outra. Assim, quando compilamos um código-fonte, estamos transformando um arquivo escrito em uma linguagem de programação em uma linguagem que o processador pode executar, também chamada de linguagem de máquina. Exemplos de linguagens compiladas: C, C++, Erlang, Haskell.
   - Outras linguagens de programação são interpretadas, funcionando da seguinte forma: o código fonte é passado para o interpretador, que pega cada instrução escrita no código-fonte e a executa de forma direta, não realizando uma tradução de todo o código-fonte. Exemplos de linguagens interpretadas: Java, Python, Ruby, PHP.
   - A maior vantagem de uma linguagem compilada é a velocidade de execução. Como, nesse tipo de linguagem, o código-fonte é convertido diretamente em uma linguagem de máquina, a execução é mais rápida e eficiente quando comparada com uma linguagem interpretada especialmente em sistemas onde a complexidade é alta.
   - Já as linguagens interpretadas não precisam ser totalmente traduzidas para um código de máquina, pois cada uma das instruções é executada diretamente pelo seu interpretador. Logo quando foram

inventadas essas linguagens, a velocidade do tempo de execução, quando comparadas com as linguagens compiladas, era, sem sombra de dúvidas, muito alta e a principal desvantagem quando feita essa comparação. Porém, com a invenção da tecnologia *just-in-time* (JIT), essa desvantagem continua existindo, mas bem menor. Essa técnica consiste em realizar uma tradução dinâmica, em tempo de execução e não antes da execução. Assim, a diferença de execução entre linguagens compiladas e interpretadas diminuiu.

- Em contrapartida, um programa compilado precisa ser gerado para cada plataforma que for rodar: Windows, Linux, Mac OS. Já um programa interpretado pode rodar em qualquer máquina, já que sua "interpretação" é feita em tempo de execução, sendo que a única necessidade é ter o interpretador na máquina onde esse programa será executado.

Link

Para saber mais sobre JIT, consulte:

https://goo.gl/GjiWhs

Referências

FORBELLONE, A. L. V.; EBERSPÄCHER, H. F. *Lógica de programação*: a construção de algoritmos e estrutura de dados. São Paulo: Makron Books, 2005.

TANENBAUM, A. S. *Sistemas operacionais*. Porto Alegre: Bookman, 2000.

DICA DO PROFESSOR

Existem várias linguagens de programação no mercado, mas há uma característica muito importante que precisa ser levada em consideração na hora da escolha.

No vídeo da Dica do Professor, você terá orientações sobre compiladores e interpretadores, ferramentas que nos auxiliam na tradução de um código-fonte.

Conteúdo interativo disponível na plataforma de ensino!

EXERCÍCIOS

1) Um algoritmo pode ser considerado como um conjunto de passos para realizar determinada tarefa.

Imagine que você precisa fazer um avião de papel e propõe para um amigo que cada um escreva um algoritmo para realizar essa tarefa. Cada um escreve o seu; vocês constatam que eles são diferentes tanto na quantidade de passos como na própria descrição dos passos. Depois dessa fase, vocês trocam os algoritmos: você faz o avião com o algoritmo escrito pelo seu amigo e vice-versa. Seguindo cada passo, os dois conseguem fazer um avião de papel.

O que é possível afirmar sobre seus algoritmos?

A) Algo está errado, pois não é possível ter mais de um algoritmo para resolver a mesma tarefa.

B) Não existe algoritmo para fazer algo que não envolva tecnologia. Portanto, estes não são algoritmos.

C) Somente quem escreve um algoritmo pode interpretá-lo. Assim, cada um deveria ter feito o avião de papel com o seu próprio algoritmo.

D) Não é possível um ser humano escrever um algoritmo para fazer um avião de papel.

E) Os algoritmos escritos estão funcionando bem, visto que atingiram o objetivo: fazer um avião de papel.

2) **Imagine um jogo de adivinhação de número. O número deve ser sorteado aleatoriamente, e o usuário adivinha um valor; se ele acertar, ganha o jogo. Caso contrário, o jogo avisa se você adivinhou um número muito grande ou muito pequeno. Um possível algoritmo para esse jogo pode ser visto a seguir:**

Passo 1 – sortear um número;

Passo 2 – perguntar ao usuário qual número ele adivinhou;

Passo 3 – se for igual ao número sorteado, informar ao usuário que ele ganhou;

Passo 4 – ?

Passo 5 – ?

Escolha a opção que completa o algoritmo com os passos 4 e 5.

A) Passo 4 – se o número adivinhado for menor que o número sorteado, dar a dica para o usuário adivinhar um número menor;
Passo 5 – se o número adivinhado for maior que o número sorteado, dar a dica para o usuário adivinhar um número maior.

B) Não é possível, pois é preciso saber o número sorteado para completar o algoritmo.

C) Passo 4 – se o número adivinhado for maior que o número sorteado, dar a dica para o usuário adivinhar um número menor;
Passo 5 – se o número adivinhado for menor que o número sorteado, dar a dica para o usuário adivinhar um número maior.

D) É impossível escrever um algoritmo para esse tipo de jogo de adivinhação.

E) Passo 4 – se o número adivinhado for menor que o número sorteado, dar a dica para o usuário adivinhar um número maior;
Passo 5 – se o número adivinhado for maior que o número sorteado, dar a dica para o usuário adivinhar um número maior.

3) ***Uma variável é uma região de memória que serve para armazenar ________ que estão envolvidos(as) num programa. As variáveis são definidas com um ________ e um ________.***

Qual opção completa de forma correta essa afirmação?

A) Dados, nome e tipo.

B) Dados, nome e valor inicial.

C) Textos, tamanho e tipo.

D) Dados, valor inicial e tipo.

E) Letras, nome e tipo.

4) **Um programa pode ser executado de duas formas: na primeira, o código-fonte é traduzido para a linguagem máquina e logo em seguida executado; na segunda, cada uma das instruções é executada de forma direta, sem a necessidade de tradução completa do código-fonte.**

Em qual das alternativas estão essas duas formas de execução, respectivamente?

A) Interpretação e compilação.

B) Simplificação e execução dinâmica.

C) Compilação e tradução.

D) Compilação e interpretação.

E) Compilação e processamento.

5) Ao comparar o tempo de execução de uma tarefa por um programa de computador escrito em linguagem compilada com outro em linguagem interpretada, é possível afirmar que:

A) os programas interpretados são mais rápidos que os programas compilados.

B) não existe diferença entre o tempo de execução de programas compilados e interpretados.

C) os programas compilados são mais rápidos que os programas interpretados.

D) não é possível avaliar o tempo de execução de um programa somente sabendo como ele foi executado.

E) o tempo de execução de dois programas não pode ser mensurado para fins de análise.

NA PRÁTICA

Rafael é um estudante de medicina que estudou algoritmo e lógica de programação durante o ensino médio. Ele está em um dos vários estágios do seu curso e precisa analisar os sintomas de um paciente para chegar a um diagnóstico.

Usando seus conhecimentos de lógica de programação, Rafael desenvolve um algoritmo capaz de ajudá-lo.

Como Rafael está em estágio, ele tem o auxílio do seu professor, que gostou da ideia e o parabenizou pela ótima ferramenta de apoio para os estudantes, salientando que pode ser melhorada com o tempo e a experiência.

SAIBA MAIS

Para ampliar o seu conhecimento a respeito desse assunto, veja abaixo as sugestões do professor:

Comece a Programar: 3 Passos Iniciais

Para complementar seus estudos, assista ao vídeo, que mostra os três principais passos para começar a programar.

Conteúdo interativo disponível na plataforma de ensino!

CS Discoveries: Variables (tem legenda)

O vídeo que mostra o que é uma variável e como sua definição é importante no desenvolvimento de algoritmos.

Conteúdo interativo disponível na plataforma de ensino!

A Hora do Código

Esta página é uma iniciativa mundial que visa a levar o ensino de programação a crianças de 10 a 16 anos de forma divertida. Acesse e confira os cursos disponíveis.

Conteúdo interativo disponível na plataforma de ensino!

A importância do uso de Algoritmos no desenvolvimento intelectual e profissional

Acesse a reportagem para complementar seus estudos sobre o uso de algoritmos e o que exatamente eles são.

Conteúdo interativo disponível na plataforma de ensino!